

# Ein Pflanzenwächter für die Fensterbank

Heute möchte ich euch ein neues interessantes mehrteiliges Projekt mit dem vielseitigen und leistungsfähigen ESP 32 vorstellen. Wie bauen uns einen Pflanzenwächter für unsere heimischen Plantae. Dieser soll während wir unsere Aufmerksamkeit unseren spannenden Elektronikprojekten widmen, den Wassergehalt der Erde überwachen und uns darüber informieren, wenn die Bodenfeuchte sinkt. Dazu hat unser Pflanzenwächter eine LED-Ampel, die bei feuchter Erde grün leuchtet und bei trockener Erde über gelb rot wird. Bevor wir jedoch mit dem Projekt beginnen, müssen wir uns jedoch noch, vor dem eigentlichen Start des Projektes einige Gedanken machen. Dies betrifft vorallendigen den Einsatz des Pflanzenwächters. Da unser Pflanzenwächter die Feuchtigkeit der Erde über eine kapazitive Nahfeldmessung bestimmt, ist es notwendig, dass die Feuchtigkeit in unmittelbarer Nähe des Sensors gespeichert wird. Während i.d.R. normale (Blumen-)Pflanzenerde aus dem Baumarkt oder dem Supermarkt diese Bedingung erfüllt, ist dies bei u.a. Hydrokultursubstraten oder Orchideensubstrat nicht der Fall! Daher:

Dieses Projekt ist für Hydrokulturlpflanzen oder Luftwurzlerpflanzen (wie z.B. Orchideen) nicht geeignet!

Außerdem haben Pflanzen ganz unterschiedliche Anforderungen an Ihre Bewässerung. Während manche Pflanzen eine dauerhafte Grundfeuchte (meistens keine Staunässe) bevorzugen, sind manche dagegen eher trockenliebend und möchten nur selten gegossen werden. Da die individuellen Anforderungen der Pflanze unser Pflanzenwächter nicht kennen kann, liegt die Interpretation der nötigen Aktionen (gießen oder nicht gießen) auf die (Ampel-) Anzeige des Pflanzenwächters ausschließlich in der Hand des botanisch kundigen Anwenders ☺.

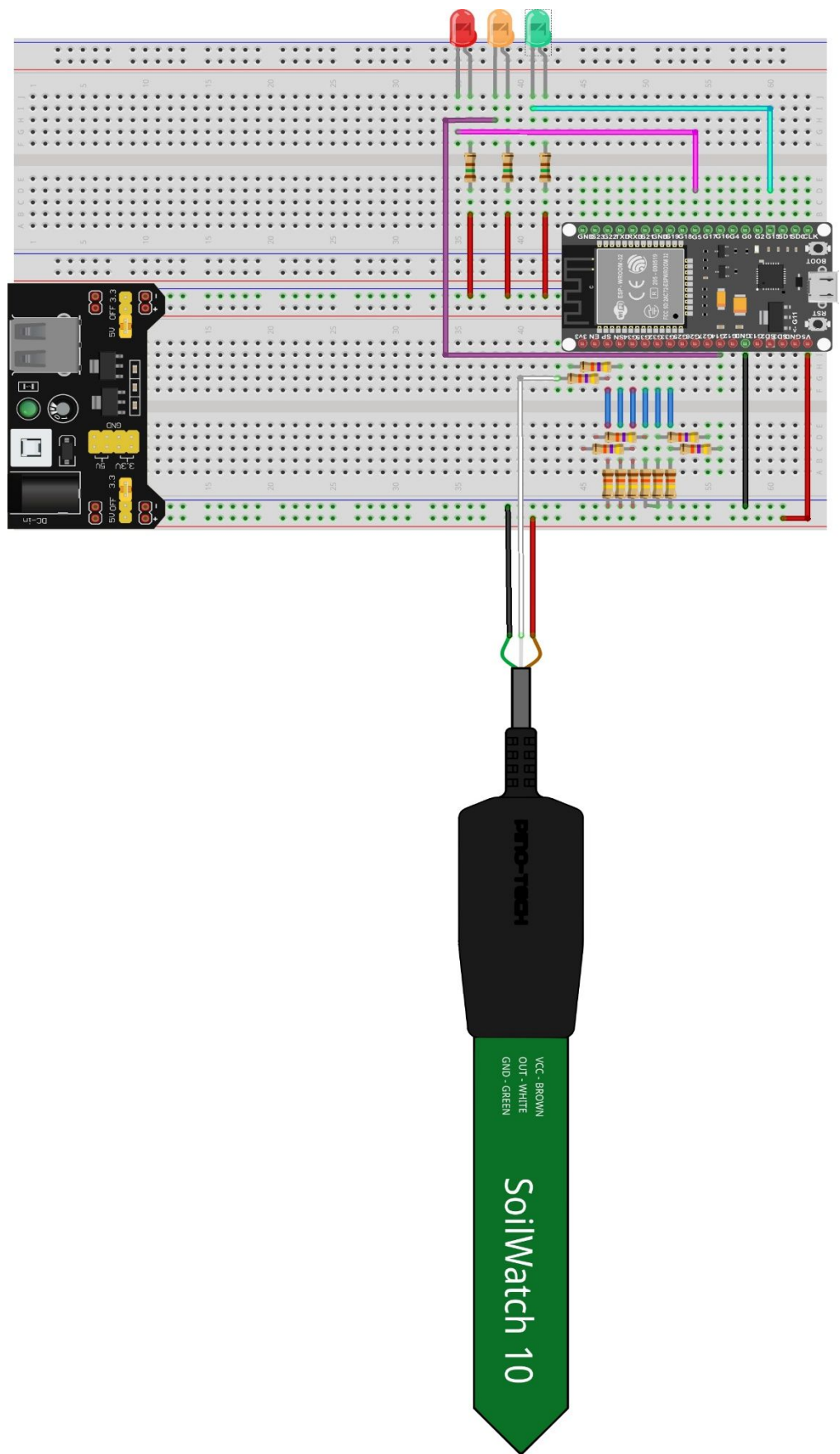
Der Pflanzenwächter ist daher kein Ersatz für eine verantwortungsvolle- und pflanzengerechte Pflege deiner Pflanzen!

Im weiteren Verlaufe des Projektes und mit größer werden dem Umfang werden wir weitere Sensoren anbinden und natürlich auch Komfortfunktionen hinzufügen. Lasst euch überraschen!

Aber fangen wir mit der Basis an. Schauen wir uns die Bauteile an, die wir für unseren Pflanzenwächter für den Anfang brauchen:

- 1 x LED Farbe Grün (560nm); 5 mm
- 1x LED Farbe Gelb (605nm); 5 mm
- 1x LED Farbe Rot (633nm); 5 mm
- 6x130kΩ Widerstand Toleranz ±1%;
- 6x 47kΩ Widerstand Toleranz ±1
- 3x 150Ω Widerstand Toleranz ±1%;
- 1x Kapazitiver Feuchtesensor
- 1x ESP32-38Pin Variante Generic; Typ NodeMCU-32S; Beinchen 38;
- 1x YwRobot Breadboard Spannungsversorgung

Wir verdrahten die Komponenten wie folgt:



Die 150 Ohm Widerstände werden als Vorwiderstände für die LEDs genutzt. Der 130 KOhm bildet zusammen mit dem 47 KOhm Widerstand ein Spannungsteiler für den Analogausgang des Feuchtigkeitssensors.

Wir laden folgenden Code auf unseren ESP 32 hoch:

```
#include <driver/adc.h>

// Portedefinierung Led's
#define LED_Rot    5    // Rote LED
#define LED_Gelb   14   // Gelbe LED
#define LED_Gruen  15   // Gruene LED

// LED PWM Einstellungen
#define PWMfreq 5000 // 5 Khz basisfrequenz
#define PWMledChannelA 0
#define PWMledChannelB 1
#define PWMledChannelC 2
#define PWMresolution 8 // 8 Bit Resolution
#define ADCAttenuation ADC_ATTEN_DB_11 //ADC_ATTEN_DB_11 = 0-3,6V
Dämpfung ADC Einstellung
#define MaxSensors 1

struct MoistureSensorCalibrationData
{
    int Data[MaxSensors*2] = {0,0}; // Calibration Data für Feuchtigkeitssensor. Bitte
    Projekt Text beachten und Werte entsprechend anpassen
};

struct MoistureSensorData
{
    byte Percent[MaxSensors] = {0}; // Feuchtigkeitssensordaten in Prozent
};

//Global Variables
MoistureSensorCalibrationData MCalib;
MoistureSensorData MMeasure;
byte AttachedMoistureSensors; // Detected Active Moisture Sensors (Count)

void setup() {
    // initialize serial communication at 9600 bits per second:
    pinMode(LED_Rot,OUTPUT);
    pinMode(LED_Gelb,OUTPUT);
    pinMode(LED_Gruen,OUTPUT);
    Serial.begin(115200);
    ledcSetup(PWMledChannelA, PWMfreq, PWMresolution);
    ledcSetup(PWMledChannelB, PWMfreq, PWMresolution);
    ledcSetup(PWMledChannelC, PWMfreq, PWMresolution);
    ledcAttachPin(LED_Rot, PWMledChannelA); // attach the channel to the GPIO
    to be controlled
    ledcAttachPin(LED_Gelb, PWMledChannelB);
```

```

    ledcAttachPin(LED_Gruen, PWMledChannelC);
    SetLedConfig(20,20,20);
    AttachedMoistureSensors = DetectMoistureSensors();
    Serial.println(F("Systemkonfiguration:"));
    Serial.print(AttachedMoistureSensors);
    Serial.println(F(" Bodenfeuchtigkeitsensor(en)"));
}

byte DetectMoistureSensors ()
{
    #define MinSensorValue 100
    byte Detected = 0;
    for (int i = 0; i < MaxSensors; i++)
    {
        int MSensorRawValue = ReadMoistureSensorVal(i);
        if ( MSensorRawValue > MinSensorValue) { Detected++; } else {break;}
    }
    if (Detected < 1)
    {
        Serial.println(F("Keine Bodenfeuchtigkeitssesoren erkannt. System
angehalten."));
        esp_deep_sleep_start();
        while(1) {}
    }
    return Detected;
}

bool SetLedConfig(byte Red, byte yellow, byte green)
{
    ledcWrite(PWMledChannelA, Red); // Rote LED
    ledcWrite(PWMledChannelB, yellow); // Gelbe LED
    ledcWrite(PWMledChannelC, green); // Gruene LED
    return true;
}

int ReadMoistureSensorVal(byte Sensor)
{
    int ReturnValue, i;
    long sum = 0;
    #define NUM_READS 6
    adc1_config_width(ADC_WIDTH_BIT_12); //Range 0-4095
    switch (Sensor)
    {
        case 0:
        {
            adc1_config_channel_atten(ADC1_CHANNEL_0, ADCAttenuation);
            for (i = 0; i < NUM_READS; i++){ // Averaging algorithm
                sum += adc1_get_raw( ADC1_CHANNEL_0 ); //Read analog
            }
            ReturnValue = sum / NUM_READS;
            break;
        }
    }
}

```

```

    }
    }
    return ReturnValue;
}

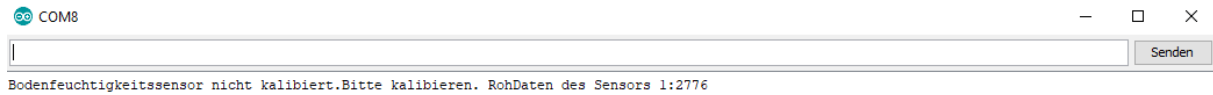
bool GetMoistureData()
{
    bool ReadisValid = true;
    for (int i = 0; i < AttachedMoistureSensors; i++)
    {
        if ((MCalib.Data[i] == 0) || (MCalib.Data[i+1] == 0)) // MinADC Value maxADC
        ADC Value
        {
            ReadisValid = false;
            return ReadisValid;
        }
        int RawMoistureValue= ReadMoistureSensorVal(i);
        RawMoistureValue= MCalib.Data[i+1] - RawMoistureValue;
        RawMoistureValue=MCalib.Data[i] + RawMoistureValue;
        MMeasure.Percent[i] = map(RawMoistureValue,MCalib.Data[i],MCalib.Data[i+1],
0, 100);
        if (MMeasure.Percent[i] > 100 ) { ReadisValid = false; }
    }
    return ReadisValid;
}

// Main Loop
void loop()
{
    if (GetMoistureData())
    {
        Serial.print(F("Feuchtigkeitswert Sensor 1 in Prozent :"));
        Serial.print(MMeasure.Percent[0]);
        Serial.println(F(" %"));
        if (MMeasure.Percent[0] > 50)
        { SetLedConfig(0,0,20); }
        else if (MMeasure.Percent[0] > 10)
        { SetLedConfig(0,255,0); }
        else
        { SetLedConfig(255,0,0); }
    }
    else
    {
        Serial.print(F("Bodenfeuchtigkeitssensor nicht kalibriert.Bitte kalibrieren. RohDaten
des Sensors 1:"));
        Serial.println(ReadMoistureSensorVal(0));
        SetLedConfig(20,20,20);
    }
    delay(1000);    // delay between reads for stability
}

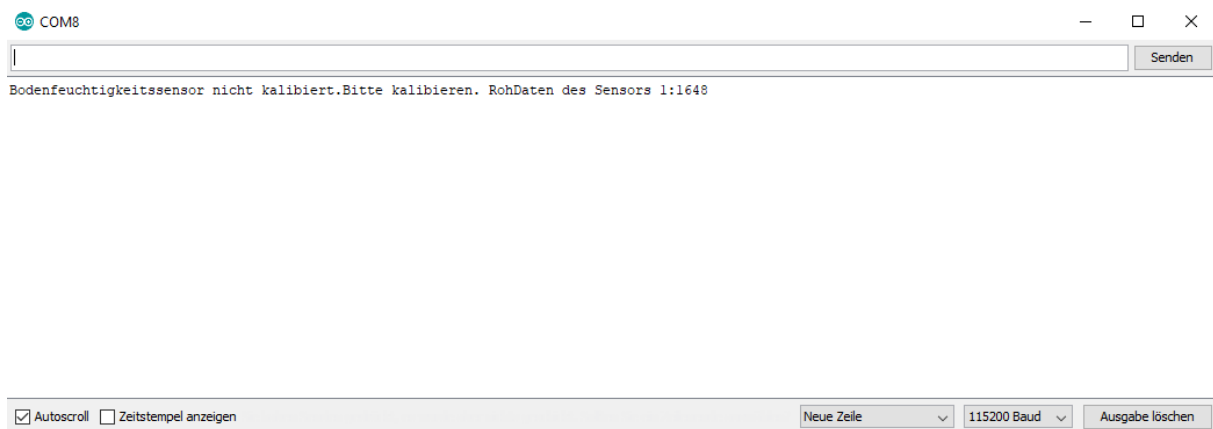
```

Als letzten Schritt müssen wir nun die Kalibrierung unseres Feuchtigkeitssensors durchführen. Die Kalibrierung des Sensors legt fest, was als trockene Erde (Wassergehalt 0%) und was nasse Erde (Wassergehalt 100%) erkannt wird. Dazu strecken wir den Feuchtigkeitssensor als erstes in absolut trockene Erde und lassen uns die Sensor- Rohdaten auf der seriellen Schnittstelle ausgeben:

Wert 1: (Trocken)



Wir notieren uns den Wert (2276) und bewässern nun die Erde solange, bis sie komplett! durchnässt ist und kein Wasser mehr ausnehmen kann. Wir notieren uns den Wert 2: (Nasse Erde) (1648)



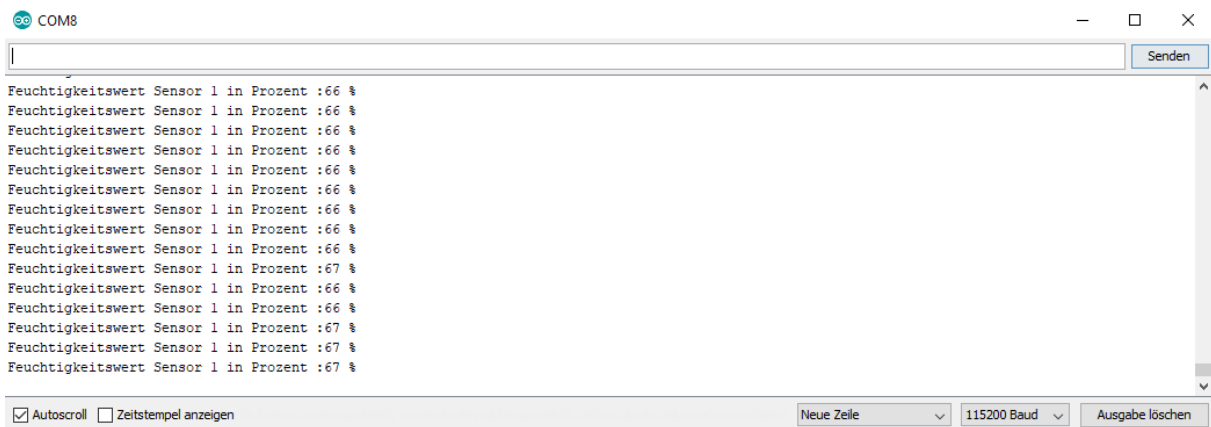
Wir addieren zu dem ersten Wert 2276, 10 dazu und ziehen von 1648, 10 ab. Es ergeben sich daraus die Werte 2286 und 1638

Wir tragen die Werte in unseren Code ein:

```
struct MoistureSensorCalibrationData
{
    int Data[MaxSensors*2] = {1638,2286};
};
```

Und laden den Code erneut hoch.

Wir erhalten folgende Ausgabe:



```
COM8
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :67 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :66 ↵
Feuchtigkeitswert Sensor 1 in Prozent :67 ↵
Feuchtigkeitswert Sensor 1 in Prozent :67 ↵
Feuchtigkeitswert Sensor 1 in Prozent :67 ↵
```

☒ Autoscroll ☐ Zeitstempel anzeigen Neue Zeile 115200 Baud Ausgabe löschen

Gleichzeitig zeigt unsere Led „Ampel“ grün an. Dabei bedeuten die Farben:

**Grün:** Feuchtigkeit hoch.

**Gelb:** Feuchtigkeit mittelmäßig.

**Rot:** Trocken.

Viel Spaß beim Nachbauen und bis zum nächsten Teil der Serie.